

AN ANALYSIS OF CHANGES DURING
MAINTENANCE OF A C SOFTWARE SYSTEM

by

KYUNG HEE AN

B.S., Korea University, 1977

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

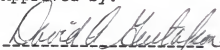
MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1987

Approved by:



Major Professor

10
1668
T4
MSC
987
15
2

TABLE OF CONTENTS

LISTS OF TABLES	ii
Chapter 1. OVERVIEW	1
Chapter 2. DATA COLLECTION AND ANALYSIS	4
2.1 Developing Definitions	4
2.2 Choosing C Modules	5
2.3 Writing Programs	6
2.4 Analyzing Data by Using the Statistic Packages	6
Chapter 3. DATA ANALYSIS	9
Chapter 4. THE RESULTS	14
4.1 Lines of Code	16
4.2 Weight	17
4.3 Average Nesting Level	19
4.4 Module Changes	20
4.5 Predictors	22
Chapter 5. CONCLUSIONS	23
Chapter 7. FUTURE WORK	27
REFERENCES	28
APPENDIX A. SHELL PROGRAMS	30
APPENDIX B. RAW DATA(123 cases and 18 variables)	35
APPENDIX C. STATEMENT TYPES	38

LISTS OF TABLES

1. The Data Analysis Table	10
2. Changes in Lines of Code	16
3. Changes in Weight	17
4. Changes in Average Nesting Level	19
5. Changed versus Not Changed Modules	20
6. Predictor Table	22

ACKNOWLEDGEMENTS

I would like to express my appreciation to Dr. David A. Gustafson for his dedicated assistance in the preparation of this thesis, and to my parents, Dalsoo & Jeungsul An, who supported me while studying in the United States.

Chapter 1. OVERVIEW

Software maintenance of computer systems is an essential task. Maintenance is a difficult and expensive activity; more time and money are spent in software maintenance than in software development. Current demands require the development of good tools for evaluating software during maintenance. The maintenance tasks would be simplified by knowing which modules are most susceptible to change and which ones should be rewritten.

Literature Survey

The maintenance process has not been sufficiently well explored. Some methods exist for predicting development characteristics of the maintenance process. No single technique can hope to solve the maintenance problem which will remain a challenge to produce greater flexibility and longer life-cycles[PARR:Pa79]. By given explicit attention to characteristics of both software quality and requirements for long-term maintenance we can produce significant savings in software lifecycle costs[BROWN:Bo76 and GILB:Gi79]. Application of improved development techniques has emphasized the need for improved techniques for requirements analysis and specification. The use of these techniques and their relationship to each other are not often clear[FREEMAN:Fr79], and the need for continuing maintenance and change of software is not primarily due to a lack of foresight or to poor planning[LEHMAN:Le79]. An understanding

of the maintenance process might be based on working with real software by improving the efficiency of the maintenance [PARR:Pa79]. The quantitative approach provokes us to ask better questions about the known effects of the various technological alternatives[GILB:Gi79].

There are two different approaches for assessing software maintainability. One is based on the extent that program difficulty represents the sum of the difficulties of its constituent elements of software[BERNS:Be84 and HALSTEAD:El76]. The other is based on a quantitative evaluation of software quality by collecting experience data in a form suited to our future and common needs[GILB:Gi79 and BOEHM:Bo76]. In the former case, the elements of software are quantified by attributes and interrelationships for checking the program difficulty or understandability rather than usability, reliability, and modifiability. In the latter case, the collection of empirical data from ongoing maintenance processes in order to measure the nature of the software is needed. The nature of maintenance work suggests that empirical analyses are the most appropriate in leading us to a greater understanding of the structure of large software systems [PARR:Pa79 and HENDERSON: He79]. These analyses may form one of the formal methodologies for the development of quality software[A B Marmor-Squires:Ma79].

This research is based on the development of a maintenance measure which was specified in the software measures research by Dr. Gustafson. The research has two fields of study. One is to develop a maintenance theory of changes and derive the method of

predicting changes from the theory. The other is to develop a maintenance measure that depends upon the empirical data of software changes. The empirical data will be obtained from the source code. This study uses the latter approach. The maintenance measure of this study could help the maintenance tasks.

We conducted an experiment to investigate changes between System3 and System5 of Unix. The System3 is the older version, and the System5 is the newer version which was created from System3. The experiment was to analyze the differences between the two versions. All the C modules were processed by our analysis programs. The differences were studied as changes to the older version. The changes were analyzed using several statistical packages to find relationships among the changes. The results support the development of a maintenance tool that could be used to predict the modules most likely to be changed. The ability to predict where changes will occur during maintenance and enhancement could minimize the extent of changes and reduce the maintenance cost.

This thesis includes an explanation of the data collection and analysis, discussion of results, an interpretation of results, a statement of conclusions, and suggestions for future extensions.

Chapter 2. DATA COLLECTION AND ANALYSIS

The first step of our research was to analyze the relation of changes between the C modules of Unix System3 and System5. The analysis performed in this research was aimed at better understanding maintenance and at developing predictive methods for the maintenance process. The sequence of the analysis was as followings; First, definitions about changes were developed. Second, 123 pairs were chosen, consisting of 35,464 lines of code in System3 and 46,023 lines of code in System5. Third, programs were written to collect all the information. Forth, the empirical data were analyzed by using several statistical packages.

2.1 Developing Definitions

Some specialized definitions were developed for the analysis according to the software measures research. The definitions for the possible changes of the modules were developed and evaluated. The chosen definitions are given below:

1) changes[type] : number of statements of specified type that have been changed.

2) change percent[type] : percentage of type that has changed.

$$\text{change percent} = (\text{changes}[\text{type}] / \text{total number}) * 100$$

3) average nesting level : the average level of nesting for the statements in a module.

$$\text{average nesting level} = (\text{SUM } \{i = 0 \text{ to } n\} \text{ of } i * \text{nli}) / \text{LOC}$$

nli : the number of statements at nesting level i
LOC : lines of code
SUM : summation

4) weight[type] : the number of statement changes for each statement type for initial study[Gu85].

$\text{weight} = \text{SUM } \{i = 0 \text{ to } k\} \text{ of } (\text{weight}[\text{type}] * x_i)$

x_i : the number of occurrence of statement type

5) $\text{weight} / \text{LOC}$: the number of statement changes per line.

Those terms were used to assess the modules for our empirical data in terms of the lines of code (LOC), weight, weight per line, and average nesting level. The lines of code shows how many lines a module has or how big it is. The weights represent the change percentages for the program statements in each module. The $\text{weight}[\text{type}]$ of each statement type was measured in the original research paper [Gu85] for seeing what statement types are most likely to be changed. The weight is quantified by the number of each statement based upon the change percent. So the weight was used as a possible measure for predicting further changes. The average level of nesting for the statements in a module is determined by the indented tabs of each line. The average nesting level represents the indented levels per line for checking how much a module is nested.

2.2 Choosing C Modules

All the C modules between System3 and System5 which had the same names in both directories were processed. Other modules were considered as improvements or changes of the system capabilities and not as merely maintenance. The total number of C modules in each system was 140. 123 modules were chosen among those for our empirical study. The other seventeen modules in System3 did not have counterparts in System5.

2.3 Writing Programs

For the analysis, several programs were written as shell programs in Unix. The programs were designed to collect the information for the different stages of each module; First, the indentation program [Appendix A.1] counts tabs of each line and calculates the average tabs which is defined as an average nesting level. Each line is classified into level zero to six based on the number of tabs. Second, the program[Appendix A.2] for nesting levels searches all the program reserved words and gathers the word counts for each module. The other part of the program[Appendix A.3] quantifies the weight according to the counts. The main program generates the lines of code, weight, and weight per line. A processing time of five hours was needed to execute the two whole directories of Unix with these programs.

2.4 Analyzing Data by Using the Statistic Packages

Three statistical packages were used to analyze our empirical data which obtained from the C modules between System3 and System5. We used three steps to start the analysis of the data. The processing steps are as followings:

First step; The possible relations of our specialized terms were expanded and all the values were calculated by the shell programs. 18 variables and 123 cases were created for the next processing. The empirical data were manipulated by the Excel system which is an advanced worksheet package for the Macintosh.

Second step; We used the Macspin for finding what sort of relations exist. Macspin is a statistical analysis

tool which is designed for high performance interaction with multivariate data. We checked all the relations with our statistical data by using graphical displays. We could find some relations and estimate the patterns through the three dimensional scatterplots of the data. The second processing step gave us the possible relationships that were quantified in step three.

Third step; The 18 variables and 123 cases(modules) were analyzed by Statfast which is a general statistical package. The package performed the statistical procedures such as t-test, student F-test, correlations, and multiple regression. The multiple regression was used to perform relative analyses on the data. This analysis allowed us to see the means, standard deviations, and the correlation matrix. The matrix of correlations were displayed as a table with 18 by 18 variables. We could observe the minimum relationships with the table. Performing the stepwise regression analyses, F-test for one dependent variable and t-test for several independent variables were evaluated based on the hypothesis tests to determine whether or not we can be reasonably confident that variables are related. The multiple regression analysis was used to know which variables will be strong predictors among several independent variables by the tests.

We obtained valuable results by repeating this step for different dependent variables. All the results were processed through the three steps.

Footnotes

- * Excel, a registered trademark of Microsoft Corporation, is a spreadsheet product for Macintosh that provides database and graphic functions, and designed for numerical processing applications.
- * Macspin, a registered trademark of D2 Software, Inc., Austin, Texas, is a tool for enable for looking at three and higher dimensional data and displays abstract multivariate data in a direct way. Its display can reveal striking patterns and relationships.
- * Statfast, a registered trademark of StatSoft, Inc., is a high performance statistical package developed in FORTRAN (MacFortran, Absoft, Inc) and offers the speed for performing statistical analysis that makes it fully suitable for scientific and business applications.
- * Macintosh, a trademark licenced to Apple Computer, Inc., is a 32 bit micro computer has powerful 68000 CPU.
- * UNIX is a registered trademark of AT&T.

Chapter 3. DATA ANALYSIS

The data analysis was performed by the three statistical packages. The source data[Appendix B] were obtained from the source codes between two directories of Unix. The data were evaluated by the packages through the specialized capabilities such as numerical processing, graphical analysis, and statistical analysis. The dependent variables of System5 were compared with the independent variables of System3 for the analysis[Table 1]. The relative relations between a dependent and independent variables were tested by the multiple regression. The hypothesis test was used to determine whether an independent variable is acceptable or not. For example, the first dependent variable in Table 1 is lines of code of the System5 and the independent variables consist of lines of code, weight, weight per line, and average nesting level of the system3. The lines of code in System5 was highly correlated with the lines of code(99.99%) and weight(88.6%) of System3. The positive relationship for lines of code and the negative relationship for the weight suggest that we can predict both relationships. But the low percentage of significant levels of the weight per line(28.4%) and average nesting level(57.4%) imply that we can not predict the relationships because of the lack of significances. The explanatory evaluations for each variable are given in the result section.

Table 1 The Data Analysis Table

Dependent Variables		Independent Variables			
(123 modules of each)		LOC3	WE3	WE/LOC3	Av.nst3
LOC5 =====	Relationships	pos	neg	pos	neg
confidence: 99.99%	Significance(%)	99.99	88.6	28.4	57.4
correlation: 0.7447	F: 34.2 > 4.95	99.99	88.6	28.4	57.4
	F: 45.6 > 5.78	99.99	85.3	12.1	
	F: 69.0 > 7.32	99.99	91.7		
WE5 =====	Relationships	pos	pos	pos	neg
confidence: 99.99%	Significance(%)	95.7	13.8	23.6	50.7
correlation: 0.6623	F: 21.5 > 4.95	95.7	13.8	23.6	50.7
	F: 43.5 > 7.32	99.99			47.1
	F: 87.1 > 11.4	99.99			
WE/LOC5 =====	Relationships	pos	pos	pos	neg
confidence: 99.99%	Significance(%)	5.9	12.9	99.99	15.8
correlation: 0.8655	F: 82.1 > 4.95	5.9	12.9	99.95	15.8
	F: 166.5 > 7.32			99.95	15.3
Av_nst5 =====	Relationships	pos	neg	pos	pos
confidence: 99.99%	Significance(%)	93.4	82.2	23.4	99.99
correlation: 0.9770	F: 577.2 > 4.95	93.4	82.2	23.4	99.99
	F: 776.0 > 5.78	95.9	86.4		99.99

(continued)

LOC5-LOC3 =====	Relationships	pos	neg	pos	neg
	Significance(%)	96.2	88.6	28.4	57.4
confidence: 90.9%	F: 2.050 ~	96.2	88.6	28.4	57.4
correlation: 0.2634	F: 2.711 ~ F: 3.811 ~	97.7 97.6	91.6 91.7		52.9
(LOC5-LOC3) / LOC3	Relationships	pos	neg	pos	neg
	Significance(%)	75.7	76.8	48.4	76.9
confidence: 36.4%	F: 0.636 ~	75.7	76.8	48.4	76.9
correlation: 0.1503	F: 0.705 ~ F: 0.513 ~	66.2 64.7	68.1 68.3		70.1
WE5 - WE3 =====	Relationships	pos	neg	pos	neg
	Significance(%)	95.7	88.9	23.6	50.7
confidence: 86.9%	F: 1.809 ~	95.7	88.9	23.6	50.7
correlation: 0.2485	F: 2.404 ~ F: 3.417 ~	97.7 97.6	92.8 92.9		47.1
(WE5-WE3) / WE3 =====	Relationships	neg	pos	neg	neg
	Significance(%)	99.4	99.4	99.99	16.2
confidence: 99.99%	F: 6.87 > 4.95	99.4	99.4	99.99	16.2
correlation: 0.4470	F: 9.22 > 5.78	99.6	99.5	99.99	
(WE/LOC5) -(WE/LOC3) =====	Relationships	pos	pos	neg	neg
	Significance(%)	6.6	12.2	99.9	16.4
confidence: 99.99%	F: 6.09 > 4.95	6.6	12.2	99.9	16.4
correlation: 0.4259	F: 12.23 > 7.32 F: 24.64 > 11.4			99.9 99.9	15.8
[(WE/LOC5)- (WE/LOC3)] / (WE/LOC3)	Relationships	neg	pos	neg	neg
	Significance(%)	99.9	99.8	99.99	17.2
confidence: 99.99%	F: 9.78 > 4.95	99.9	99.8	99.99	17.2
correlation: 0.5122					

(continued)

(Av_nst5 - Av_nst3)	Relationships	pos	neg	pos	neg
=====	Significance(%)	93.4	82.2	23.3	99.9
confidence:					
99.8%	F: 4.77 > 4.50	93.4	82.2	23.3	99.9
correlation:	F: 6.38 > 5.78	95.9	86.4		99.9
0.3843	F: 8.37 > 7.32	96.8			99.9
(Av_nst5- Av_nst3)	Relationships	pos	neg	neg	neg
/ Av_nst3	Significance(%)	6.3	15.9	30.2	94.3
confidence:					
79.1%	F: 1.490 ~	6.3	15.9	30.2	94.3
correlation:	F: 2.798 ~			52.5	95.1
0.2267	F: 5.087 ~				97.6
LOC5-LOC3	Relationships	neg	pos	neg	pos
= 0 (8)					
LOC5-LOC3	Significance(%)	25.7	5.7	57.8	80.2
= 1 (115)					
	F: 1.192 ~	25.7	5.7	57.8	80.2
correlation:	F: 1.881 ~	75.8			90.6
0.2120	F: 2.379 ~				87.8

* pos is a positive relation, and neg is a negative relation.

(continued)

LOC5-LOC3 vs. WE5-WE3 vs. Av nst5-Av nst3

Dependent Variables		Independent Variables		
(123 modules of each)		WE5-WE3	WE/LOC5	Av_nst5
#: Distribution Table			WE/LOC3	Av_nst3
LOC5-LOC3	Relationships	pos	neg	pos
=====				
confidence:	Significance(%)	99.99	99.99	99.9
99.99%				
correlation:	F: 1682.35	t: 65.82	t: 6.07	t: 3.39
0.9892	# 4.95	# 3.84	# 3.84	# 3.16
Dependent		LOC5	WE/LOC5	Av_nst5
		-	-	-
		LOC3	WE/LOC3	Av_nst3
WE5-WE3	Relationships	pos	pos	neg
=====				
confidence:	Significance(%)	99.99	99.99	99.7
99.99%				
correlation:	F: 1748.12	t: 65.82	t: 6.65	t: 3.03
0.9973	# 4.95	# 3.84	# 3.84	# 2.76
Dependent		LOC5	WE5-WE3	WE/LOC5
		-		-
		LOC3		WE/LOC3
Avnst5-Avnst3	Relationships	pos	neg	pos
=====				
confidence:	Significance(%)	99.9	99.7	99.99
99.99%				
correlation:	F: 13.23	t: 3.39	t: 3.03	t: 4.89
0.5153	# 4.95	# 3.16	# 2.76	# 3.84

Chapter 4. THE RESULTS

The technique of the regression analysis was used to check on relationships between variables and also to assist in determining the best set of predictor variables. In order to evaluate some observed values among several variables, an hypothesis test was performed first. The alternative hypothesis is two sided. We tested all the predictors in order to detect those inversely related to Y axis as well as those directly related. The null and alternative hypotheses could alternatively be written in vector notation as

$$Y = A + B_1X_1 + B_2X_2 + \dots + B_kX_k + e$$

H0 : B = 0 (no relationship) : reject
H1 : B > 0 (direct relationship) : accept
H1 : B < 0 (inverse relationship) : accept

H : hypothesis
Y : linear function of k predictor variables, X_1, X_2, \dots, X_k
A : significance level
B : significance from a regression equation
e : error term

The test showed which was acceptable or rejectable in a given criteria.

In order to verify the validity of the predictors, The F-distribution and the student's t-distribution were used to test whether there were significant differences between the means of samples drawn from the normally distributed variables. Therefore, F-test was performed for a dependent variable among multiple variables with a confidence level and t-test was performed for multiple independent variables with several significant levels for the several variables.

If the F _value or t _values are acceptable, the relationships among the several variables exist. So, the regression analysis was concerned with investigating the relationships among several variables by showing which variables could be strong predictors of the response variable.

The experiment was to analyze two versions of Unix between dependent variables and independent variables. The variables were chosen among the terms which specified in the software measures research. The relative relations between a dependent and independent variables were obtained by the multiple regression test.

The percentage of the confidence level and significant level was used to determine whether the variable is reliable or not. Usually, the levels with over 75 percent will be considered reliable by most logicians and mathematicians.

The negative value of t -test implies a negative correlation between variables, and the positive value implies a positive correlation between variables. So, the variables were evaluated by the correlations.

4.1 Lines of Code

Dependent Variables		Independent Variables			
(123 modules of each)		LOC3	WE3	WE/LOC3	Av.nst3
LOC5-LOC3 =====	Relationships	pos	neg	pos	neg
	Significance(%)	96.2	88.6	28.4	57.4
confidence: 90.9%	F: 2.050 ~	96.2	88.6	28.4	57.4
correlation: 0.2634	F: 2.711 ~	97.7	91.6		52.9
	F: 3.811 ~	97.6	91.7		
(LOC5-LOC3) / LOC3	Relationships	pos	neg	pos	neg
	Significance(%)	75.7	76.8	48.4	76.9
confidence: 36.4%	F: 0.636 ~	75.7	76.8	48.4	76.9
correlation: 0.1503	F: 0.705 ~	66.2	68.1		70.1
	F: 0.513 ~	64.7	68.3		

Table 2 Changes in Lines of Code

The partial results of the multiple regression test for changes in lines of code are given in Table 2. The dependent variable is the increase in the number of lines of code and the independent variables consist of lines of code, weight, weight per line, and average nesting level of the system3. We investigated how the lines of code in System5 are related to the lines of code, weight, weight per line, and average nesting level of System3.

The increase in lines of code in System5 was highly correlated with the lines of code(96.2%) and weight(88.6%) of System3 with significant levels that were much higher than the standard cutoff point of 75 percent. The positive relationship

for lines of code implies that the larger modules will have larger increases in lines of code[section 5.1.c]. But the significant levels of the weight per line(28.4%) and average nesting level(57.4%) of System3 were below the standard cutoff point. The low percentage of significant level implies that we can not predict the relationships because of lack of significance. The relation between the updated lines and weight is surprising. The negative relation implies that the modules with higher weighting have smaller changes in lines of code[section 5.2.b]. Therefore, a module with many high risk statements will tend to decrease in lines of code during maintenance or enhancement[section 5.2].

4.2 Weights

Dependent Variables		Independent Variables			
(123 modules of each)		LOC3	WE3	WE/LOC3	Av.nst3
WE5 - WE3 =====	Relationships	pos	neg	pos	neg
	Significance(%)	95.7	88.9	23.6	50.7
confidence: 86.9%	F: 1.809 ~	95.7	88.9	23.6	50.7
correlation: 0.2485	F: 2.404 ~ F: 3.417 ~	97.7 97.6	92.8 92.9		47.1
(WE/LOC5) -(WE/LOC3) =====	Relationships	pos	pos	neg	neg
	Significance(%)	6.6	12.2	99.9	16.4
confidence: 99.99%	F: 6.09 > 4.95	6.6	12.2	99.9	16.4
correlation: 0.4259	F:12.23 > 7.32 F:24.64 > 11.4			99.9 99.9	15.8

Table 3 Changes in Weights

The dependent variable in Table 3 is the difference of weight between the two system. The investigation was how weight changes are related to lines of code, weight, weight per line, and average nesting level of System3. The difference of weight between two system was correlated with the lines of code(95.7%) and weight(88.9%) of System3 with the significant levels that were higher than the standard cutoff point. But we did not consider the relations for the weight per line(23.6%) and average nesting level(50.7%) of System3 because of their low confidence levels. The positive relation for the lines of code implies that larger modules tend to have larger increases in weights[section 5.2.a]. The negative relation for the weight implies that modules with higher weighting tend to have decreases in weights[section 5.4.a]. If the weight of a module is relatively high, it will tend to decrease during maintenance[section 5.4].

4.3 Average Nesting Level

Dependent Variables		Independent Variables			
(123 modules of each)		LOC3	WE3	WE/LOC3	Av.nst3
(Av_nst5 - Av_nst3)	Relationships	pos	neg	pos	neg
=====	Significance(%)	93.4	82.2	23.3	99.9
confidence:					
99.8%	F: 4.77 > 4.50	93.4	82.2	23.3	99.9
correlation:	F: 6.38 > 5.78	95.9	86.4		99.9
0.3843	F: 8.37 > 7.32	96.8			99.9
(Av_nst5 - Av_nst3) / Av.nst3	Relationships	pos	neg	neg	neg
	Significance(%)	6.3	15.9	30.2	94.3
confidence:					
79.1%	F: 1.490 ~	6.3	15.9	30.2	94.3
correlation:	F: 2.798 ~			52.5	95.1
0.2267	F: 5.087 ~				97.6

Table 4 Changes in Average Nesting Level

The difference of the average nesting level between System3 and System5 in Table 4 was compared with the lines of code, weight, weight per line, and average nesting level of System3. The significant levels of the lines of code(93.4%), weight(82.2%), and average nesting level(99.9%) were very high. So, the difference of the average nesting level between the two system was correlated with the lines of code, weight, weight per line, and average nesting level of System3. These independents will be strong predictors. But the weight per line(23.3%) of System3 will not be a predictor because of the low significant level. The relation between difference of average nesting level

and lines of code was positive. The positive relation implies that the larger modules will tend to have larger increases in nesting level[section 5.3.a]. In other words, if the size of code is large, the nesting levels will tend to increase more during maintenance or enhancement[section 5.3]. The negative relation for the weight implies that modules with higher weighting will have smaller increases or even decreases in average nesting level[section 5.5.a]. The other negative relation for the average nesting level implies that modules with higher nesting levels tend to have decreases or smaller increases in nesting level[section 5.6.c]. In other words, if the nesting levels are high, they will tend to be reduced or only slightly increased during maintenance[section 5.6].

4.4 Module Changes

Measures	Changed Modules	Not Changed Modules	Total
Average lines of code	279.365	417.125	288.325
Average nesting levels	68.643	51.943	67.556

Table 5 CHANGED versus NOT CHANGED MODULES

The changed and not changed modules in Table 5 were measured by quantitative analysis. We investigated which modules will not be changed and which ones will be changed. One surprise was that most of the unchanged modules were big. The average lines of

code(417.125) of the unchanged modules was greater than the average lines of code(279.365) of the changed modules. These numbers implies that the average size of a changed module is usually smaller than the average size of ones not changed[section 5.1.a]. In other words, if a module size is relatively big, it will tend not to be changed during maintenance[section 5.1.b].

The average nesting levels(68.643) of changed modules were greater than the average nesting levels(51.943) of not changed modules. The low average nesting level of the not changed modules implies that modules with lower nesting levels will tend not to be changed[section 5.6.b]. The high average number of the changed modules implies that the highly nested modules will likely to be changed during maintenance or enhancement[section 5.6.a]. Therefore, these results seem to suggest that size and nesting may be good predictors for maintenance[section 5.1, 5.6].

4.5 Predictors

The predictor table consisted of the predictors and predicted factors which had strong relationships with the lines of code, weights, and nesting levels between System3 and System5[Table 6]. The relationships were obtained through the all the analysis process. We could predict program changes in modules by using this table during maintenance or enhancements.

Table 6 Predictor Table

predicted factors predictors		Lines of code	Weights	Nesting levels	Modules will be changed or not
Lines of code	larger	increase (5.1.c)	increase (5.2.a)	increase (5.3.a)	not change (5.1.b)
	smaller	-	-	-	change (5.1.a)
Weights	higher	decrease (5.2.b)	decrease (5.4.a)	decrease (5.5.a)	-
	lower	-	increase (5.4.b)	-	-
		(weak)	(weak)		
Nesting levels	higher	decrease (5.3.b)	decrease (5.5.b)	decrease (5.6.c)	change (5.6.a)
	lower	-	-	-	not change (5.6.b)

* (5.--) represent concluding number.

Chapter 5. CONCLUSIONS

The following relationships were assessed from our empirical data for the predictor variables such as module size, lines of code, weight, and average nesting level. These variables can be used to better understanding software maintenance. The predictors could be evaluated as followings:

5.1 LOC vs. LOC

- a) The average size of a changed module is usually smaller than the average size of ones not changed.[section 4.4]
- b) If a module size is relatively big, it will tend not to be changed.[section 4.4]
- c) Larger modules tend to have larger increases in lines of code. The percentage of updated lines will be increased too. If the modules are changed, more code will be added. [section 4.1]

Changes in size of modules during maintenance will be related to the original size of the modules. We predict that the smaller modules are more likely to be changed. When a larger module is changed, it will increase more than the smaller one.

5.2 LOC vs. WE

- a) Larger modules tend to have larger increases in weights. [section 4.2]
- b) Modules with higher weighting tend to have smaller increases in lines of code or decreases of the lines of code. [section 4.1]

More high risk statements will be added to larger modules when they are updated. A module which has more high risk statements[in higher weight] will tend to be modified less.

5.3 Ave_nest vs. LOC

- a) Larger modules tend to have larger increases in nesting level.[section 4.3]
- b) Modules with higher nesting levels tend to have smaller increases or even decreases in lines of code.[section 4.1]

Average nesting levels will increase in larger modules when they are maintained. A module which has higher nesting level will tend to be modified less.

5.4 WE vs. WE

- a) Modules with higher weighting tend to have smaller increases or even decreases in weight.[section 4.2]
- b) Modules with lower weighting tend to have larger increases in weight.[section 4.2]

Modules which have more high risk statements will tend to decrease those statements when they are maintained.

5.5 WE vs. Ave.nst

- a) Modules with higher weighting tend to have decreases in average nesting level.[sectin 4.3]
- b) Modules with higher nesting levels tend to have smaller increases or even decreases in weight.[section 4.2]

Modules which have more high risk statements will tend to decrease in average nesting level when they are maintained.

5.6 Ave_nst vs. Ave_nst

- a) The highly nested modules will likely to be changed during maintenance.[section 4.4]
- b) If nesting levels are low, a module will tend not to be changed.[section 4.4]
- c) Modules with higher nesting levels tend to have smaller increases or even decreases in nesting level.[section 4.3]

Modules which have higher average nesting level will tend to decrease in average nesting level during maintenance or enhancement.

Currently, the process of maintaining software is not well understood. The maintenance task will be helped by knowing which modules are susceptible and which ones should be rewritten. The significant relationships between the source code and possible changes will be used to suggest improvements in both the program development process and the program maintenance process.

What possibilities to improve maintenance?

Our predicted maintenance approach will enable better planning and management of maintenance work. Making program modules more easily maintainable could reduce the maintenance tasks. Our approach would suggest following possibilities:

- a) Identifying some types of maintenance work.
- b) Identifying modules to be rewritten.(which ones to modify)

The modules that are the most change prone can be rewritten to improve the future maintainability of the program.

- c) Identifying normal maintenance vs. abnormal maintenance.

What advice to developers?

To solve the maintenance problems the tasks of developing software must be simplified and automated. The developers might be able to reduce the maintenance cost by trying to develop stable modules. Our advice is as following:

- a) Stabilize the size of code.
 - Larger modules will be increased more than small ones.
 - More high risk statements will be added to larger modules.

- Nesting levels will increase in larger modules.
- Relative smaller modules will be more stable than larger ones.

b) Don't discourage code with higher weighting.

- Modules with higher weighting will tend to decrease in lines of code.
- Modules with higher weighting tend to have decreases in weights.
- Module with higher weighting tend to have decreases in nesting level.
- Modules with higher weighting will tend to be stable.

c) Don't discourage highly nested code.

- Modules with higher nesting levels tend to have smaller increases in lines of code.
- Modules with higher nesting levels tend to have decreases in weight.
- Modules with higher nesting levels tend to have smaller increases in nesting level.
- However highly nested modules will be likely to change.

Chapter 7. FUTURE WORK

The results of our analyses suggest five different areas for future research:

- 1) the analysis will continue to try to find more relationships between the source code and the changes,
- 2) the analysis will be done on other systems in other languages to try to generalize the conclusions,
- 3) the relationships will be used to develop a maintenance measure to predict the possible changes according to the module size, weight, and nesting level,
- 4) further comparing the changes to systems during development and changes to systems during maintenance will be conducted and,
- 5) patterns of changes for each of the maintenance activities defined by Swanson (corrective, adaptive, and perfective maintenance) will be developed to aid in analyzing maintenance.

REFERENCES

- <Yo79> Younger, Mary Sue, Handbook for Linear Regression, Duxbury press, Wadsworth Inc., Belmont, California, 1979.
- <Wa86> Wall, Francis J., Statistical Data Analysis Handbook, McGraw-Hill Inc., 1986.
- <St80> Stoodley, K.D.C., Lewis, T., and Stainton, C.L.S., Applied Statistical Techniques, Ellis Horwood Series: Mathematics and Its Application, Halsted Press, 1980.
- <To85> Townsend, Carl, Mastering Excel, Sybex Inc., Berkeley, California, 1985.
- <Do85> Donoho, Andrew W., Donoho, David L., and Gasko, Miriam, Macspin User's Manual, D2 Software Inc., Austin, Texas, 1985.
- <So85> Sobel, Mark G., A Practical Guide to UNIX System V, The Benjamin/Cummings Publishing Company, Inc., Menlo Park, California.
- <Ri80> Ritchie, Dennis M., C Reference Manual, Bell Telephone Laboratories, Murray Hill, New Jersey.
- <Ma83> Martin, James and McClure, Carma, Software Maintenance: The Problem and Its Solutions, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1983.
- <Gi77> Gilb, Tom, "The Measurement of Software Reliability and Maintainability: Some Unconventional Approaches to Reliable Software." Computers and People, Sept. 1977, pp. 16-21.
- <Gu85> Gustafson, David A., Austin Melton and Chyuan Samuel Hsieh, "An Analysis of Software Changes During Maintenance and Enhancement", Conference on Software Maintenance, Washington D.C., Nov. 1985.
- <Bo76> Boehm, B. W., J. R. Brown, and M. Lipow, "Quantitative Evaluation of Software Quality." Proc. IEEE/ACM 2nd Int. Conf. Software Eng., Oct. 1976, pp. 592-605.
- <Be84> Berns, Gerald M., "Assessing Software Maintainability." Communications of the ACM, vol 27, no 1, (Jan.84), pp. 14-23.
- <El76> Elshoff, J. L., "An Analysis of Some Commercial PL/I Programs," IEEE Transactions on Software Engineering, June 1976, pp. 113-120.

- <Be85> Berry, R. E., and Meekings, B. A. E., "A Style Analysis of C Programs." Communications of the ACM, vol 28, no 1 (Jan.85), pp. 80-88.
- <Br75> Brantley, C. L., and Y. R. Osajima, "Continuing Development of Centrally Developed and Maintained Software Systems." IEEE Computer Society Conference Proceedings (COMPCON), Spring 1975, pp. 285-288.
- <St75> Stearns, Stephen K., "Experience with Centralized Maintenance of a Large Application System." IEEE Computer Society Conference Proceedings (COMPCON), Spring 1975, pp. 281-284.
- <Fr79> Freeman, P., "A Perspective on Requirements Analysis and Specification." Infotech State of Art Report, 1979, pp. 41-55.
- <Le79> Lehman, M. M., "The Software Engineering Environment." Infotech State of Art Report, 1979, pp. 147-164.
- <Gi79> Gilb, T., "Structured Design Methods for Maintainability." Infotech State of Art Report, 1979, pp. 85-98.
- <He79> Henderson, P., Gimson, Pratten, G. D., and Snowdon, R. A., "The Maintenance of Software with Multiple Versions." Infotech State of Art Report, 1979, pp. 99-115.
- <El76> Elshoff, j., "Measuring Commercial PL/1 Programs Using Halstead's Criteria," ACM SIGPLAN Notices, May 1976, pp. 38-46
- <Ma79> Marmor-Squires, A. B., "On Methodologies and Programming Environments to Support the Development of Verified Software." Infotech State of Art Report, 1979, pp. 165-176.
- <Pa79> Parr, F. N., "Software Maintenance." Infotech State of Art Report, 1979, pp. 227-237.

APPENDIX A : SHELL PROGRAMS

A.1 Counting Program for the Nesting Levels

```
#-----#
# Indent module : checks the nesting level. #
#-----#

awk -s '
{count = 0}
/^      / {count = 1}
/      / {count = 2}
/      / {count = 3}
/      / {count = 4}
/      / {count = 5}
/      / {count = 6}
{print count }' $1
```

A.2 Statistics for the Nesting Levels

```
#-----#
# Average nesting module: calculates the average nesting levels #
#-----#

awk -s '
BEGIN { printf "Levels :   \n" >> "total0"
        printf "          \n" >> "total0"
        zero = 0
        one = 0
        two = 0
        three = 0
        four = 0
        five = 0
        six = 0
        sum = 0
      }
/0/ {zero = zero + 1}
/1/ {one = one + 1}
/2/ {two = two + 1}
/3/ {three = three + 1}
/4/ {four = four + 1}
/5/ {five = five + 1}
/6/ {six = six + 1}
END { printf "zero   %6d\n", zero   >> "total0"
      printf "one    %6d\n", one    >> "total0"
      printf "two    %6d\n", two    >> "total0"
      printf "three  %6d\n", three  >> "total0"
      printf "four   %6d\n", four   >> "total0"
      printf "five   %6d\n", five   >> "total0"
      printf "six    %6d\n", six    >> "total0"
      printf "-----\n" >> "total0"
      zeroave = (zero * 100) / NR
      printf "Zeroave =   %5.3f\n", zeroave   >> "total0"
```

(continued)

```
oneave = (one * 100) / NR
printf "Oneave =      %5.3f\n", oneave      >> "total0"
twoave = (two * 100) / NR
printf "Twoave =      %5.3f\n", twoave      >> "total0"
threeave = (three * 100) / NR
printf "Threeave =    %5.3f\n", threeave    >> "total0"
fourave = (four * 100) / NR
printf "Fourave =     %5.3f\n", fourave     >> "total0"
fiveave = (five * 100) / NR
printf "Fiveave =     %5.3f\n", fiveave     >> "total0"
sixave = (six * 100) / NR
printf "Sixave =      %5.3f\n", sixave      >> "total0"
average = 100 * (zero + one*2 + two*3 + three*4) / NR
average += 100 * (four*5 + five*6 + six*7) / NR
printf "Total average = %5.3f\n", average  >> "total0"
sum = zero + one + two + three + four + five + six
printf "Sum =          %10d\n", sum        >> "total0"
printf "Lines of code = %10d\n", NR        >> "total0"
printf "Sum/Lines :     %10.3f\n", (sum/NR) >> "total0"
printf "-----\n" >> "total0"
printf "                \n" >> "total0"
printf "                \n" >> "total0"
printf "                \n" >> "total0"
printf "                \n" >> "total0"
}'
```

A.3 Weights

```
#-----#
# Weight module: calculates the weight of each source program #
#-----#

BEGIN { CommentSw = 0; LineNumber = 0 }
{
  #
  # process all the number of fields in the current record.
  #
  i = 1
  while (i <= NF)
  {
    if ((LineNumber + 2) <= NR)
    {
      count["blanklines"]=count["blanklines"]+NR-(LineNumber+1)
      LineNumber = NR
    }
    #
    # check the comment switch true.
    #
    if (CommentSw == 1)
    {
      if ($i == "*/")
        CommentSw = 0
    }
  }
}
```

(continued)

```
# if the comment switch is false.

else {
    if ($i == "/*")
    {
        CommentSw = 1
        count["comments"]++
    }
    # if the current field is not a comment field.
    else {

if (((($1 ~ /\:/) || ($2 ~ /\:/)) && ($i == $1))
{
    if ($1 ~ /default/)      count["default"]++
    else if ($1 != "case")   count["labels"]++
}
if ($i ~ /\(/)
{
    # split the source line delimited by "(.
    NoOfElement = split($i, Array, "(")

    # check functions inside the 'if, while, for...'

    count["functions"] = count["functions"] + NoOfElement - 1

for (k=1; k <= NoOfElement; k++)
{
    if (Array[k] == "if")
    {count["if"]++      count["functions"]--}
    else if (Array[k] == "for")
    {count["for"]++     count["functions"]--}
    else if (Array[k] == "while")
    {count["while"]++   count["functions"]--}
    else if (Array[k] == "switch")
    {count["switch"]++  count["functions"]--}
    else if (Array[k] == "rerturn")
    {count["return"]++  count["functions"]--}
    else if ((Array[k] == "getchar") || (Array[k] == "getc"))
    {count["input"]++   count["functions"]--}
    else if (Array[k] == "scanf")
    {count["input"]++   count["functions"]--}
    else if ((Array[k] == "putchar") || (Array[k] == "putc"))
    {count["output"]++  count["functions"]--}
    else if (Array[k] == "printf")
    {count["output"]++  count["functions"]--}
    else if (Array[k] == "printf")
    {count["output"]++  count["functions"]--}
}
} # end 'if ($i ~ /\(/)'

if ($i ~ /\=/)
{
```

(continued)

```
if ((($i !~ /\|=/) && ($i !~ /\==/) && ($i !~ /\<=/) && ($i !~ /\>=/))
{
    count["assignments"]++
}
# end 'if ($i ` /\|=/)'

# ..... declarations .....
    if ($i == "int")        count["declarations"]++
    else if ($i == "float") count["declarations"]++
    else if ($i == "double") count["declarations"]++
    else if ($i == "struct") count["declarations"]++
    else if ($i == "auto")   count["declarations"]++
    else if ($i == "extern") count["declarations"]++
    else if ($i == "register") count["declarations"]++
    else if ($i == "static") count["declarations"]++
    else if ($i == "if")     {count["if"]++
                             count["functions"]--}
    else if ($i == "for")    {count["for"]++
                             count["functions"]--}
    else if ($i == "while")  {count["while"]++
                             count["functions"]--}
    else if ($i == "switch") {count["switch"]++
                             count["functions"]--}
    else if (($i == "return") || ($i == "return;"))
    {
        count["return"]++
        if (($i == "return") && ($i+1) ~ /\(/))
            count["functions"]--}
    else if (($i == "getchar") || ($i == "getc"))
        {count["input"]++
         count["functions"]--}
    else if ($i == "scanf")  {count["input"]++
                             count["functions"]--}
    else if (($i == "putchar") || ($i == "putc"))
        {count["output"]++
         count["functions"]--}
    else if ($i == "printf") {count["output"]++
                             count["functions"]--}
    else if ($i == "printw") {count["output"]++
                             count["functions"]--}
    else if ($i == "else")   count["else"]++
    else if ($i ~ /\#/ )    count["preprocessor"]++
    else if ($i == "case")   count["case"]++
    else if ($i == "goto")   count["goto"]++
    else if (($i == "break") || ($i == "break;"))
        count["break"]++
    else if (($i == "continue") || ($i == "continue;"))
        count["continue"]++
    }
}
LineNumber = NR
++i
}
```

(continued)

```
END {
printf "=====\n" >> "total0";
printf "File Name : " FILENAME "\n" >> "total0";
printf "=====\n" >> "total0";
printf "for %10d\n", count["for"] >> "total0";
printf "while %10d\n", count["while"] >> "total0";
printf "if %10d\n", count["if"] >> "total0";
printf "else %10d\n", count["else"] >> "total0";
printf "switch %10d\n", count["switch"] >> "total0";
printf "case %10d\n", count["case"] >> "total0";
printf "goto %10d\n", count["goto"] >> "total0";
printf "break %10d\n", count["break"] >> "total0";
printf "continue %10d\n", count["continue"] >> "total0";
printf "assignments %10d\n", count["assignments"] >> "total0";
printf "preprocessor %10d\n", count["preprocessor"] >> "total0";
printf "comments %10d\n", count["comments"] >> "total0";
printf "blanklines %10d\n", count["blanklines"] >> "total0";
printf "return %10d\n", count["return"] >> "total0";
printf "input %10d\n", count["input"] >> "total0";
printf "output %10d\n", count["output"] >> "total0";
printf "functions %10d\n", count["functions"] >> "total0";
printf "declarations %10d\n", count["declarations"] >> "total0";
printf "default %10d\n", count["default"] >> "total0";

# calculate the weights

weights = 18.4 * count["declarations"] + 11.4 * count["if"]
weights += 7.9 * count["for"] + 8.5 * count["while"]
weights += 6.8 * count["switch"] + 5.6 * count["case"]
weights += 4.6 * count["preprocessor"] + 11.1 * count["goto"]
weights += 2.4 * count["comments"]

printf "-----\n" >> "total0"
printf "Weights = %10.5f\n", weights >> "total0"
printf "Lines of code = %10d\n", NR >> "total0";
printf "Weights/Lines = %10.5f\n", (weights/NR) >> "total0";
printf "-----\n" >> "total0";
}
```

APPENDIX B. RAW DATA(123 cases and 18 variables)

C Unix Programs (123)	1. LOC3	2. WE3	3. WE3	4. WE3	5. WE3	6. WE3	7. WE3	8. WE3	9. WE3	10. (LOC3-WE3)	11. LOC3-LOC3	12. (LOC3-LOC3)/LOC3	13. (WE3-WE3)/WE3	14. (WE3-WE3)/WE3	15. (WE3-WE3)/WE3	16. (WE3-WE3)/WE3	17. (WE3-WE3)/WE3	18. (WE3-WE3)/WE3
Names	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
400.c	443	1316.0	2.97	49.44	443	1316.0	2.97	49.44	154.67	23924.29	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00
300.c	2.97	49.44	131.4	2.97	50.11	152.67	23909.59	0	0.00	-18.4	1.38	-0.04	1.38	-0.04	1.38	-0.04	1.38	0.00
4014.c	316	1122.6	3.60	90.41	317	1063.5	2.75	27.20	97.67	690.37	0	0.68	0.00	-0.05	0.00	0.36	0.13	0.14
450.c	386	1061.5	2.75	27.20	386	1063.5	2.75	27.20	97.67	690.37	0	0.68	0.00	-0.05	0.00	0.36	0.13	0.14
arcv.c	219	281.7	2.64	3.67	438	875.2	2.00	46.80	-179.33	32157.53	339	301.83	587.5	204.20	-0.64	24.20	43.13	1172.0
banner.c	219	281.7	2.64	3.67	438	875.2	2.00	46.80	-179.33	32157.53	339	301.83	587.5	204.20	-0.64	24.20	43.13	1172.0
bcopy.c	113	439.2	3.89	38.05	119	441.6	3.71	36.13	-175.33	30738.93	6	5.31	2.4	0.55	-0.18	4.51	-1.92	5.63
bif.c	1255	5351.3	4.26	44.78	1278	5425.1	4.24	44.05	966.67	934460.19	23	1.83	73.8	1.38	-0.02	0.44	-0.73	1.63
cal.c	204	470.7	2.31	30.88	205	493.7	2.41	30.73	-84.33	7110.74	1	0.49	23.8	4.89	0.10	4.36	-0.15	0.49
cb.c	376	1358.7	3.61	39.48	376	1358.7	3.61	39.48	-107.33	17686.70	763	202.53	64.2	15.47	-0.05	1.17	-7.88	7.18
cb.c	376	1358.7	3.61	39.48	376	1358.7	3.61	39.48	-107.33	17686.70	763	202.53	64.2	15.47	-0.05	1.17	-7.88	7.18
checkkw.c	181	559.1	3.09	166.85	182	589.1	3.20	166.90	-187.33	35090.70	1	0.99	23.0	6.83	0.15	5.72	-0.92	0.55
checkkw.c	181	559.1	3.09	166.85	182	589.1	3.20	166.90	-187.33	35090.70	1	0.99	23.0	6.83	0.15	5.72	-0.92	0.55
chgrp.c	53	209.2	3.95	37.74	54	211.6	3.92	37.04	-235.33	55377.95	1	1.89	2.4	1.15	-0.03	0.72	-0.70	1.85
chgrp.c	53	209.2	3.95	37.74	54	211.6	3.92	37.04	-235.33	55377.95	1	1.89	2.4	1.15	-0.03	0.72	-0.70	1.85
chown.c	175	519.4	2.97	54.29	176	521.8	2.96	53.98	-113.33	12842.60	1	0.57	2.4	4.06	0.00	0.11	-0.31	0.57
chown.c	175	519.4	2.97	54.29	176	521.8	2.96	53.98	-113.33	12842.60	1	0.57	2.4	4.06	0.00	0.11	-0.31	0.57
chroot.c	77	55.0	2.04	25.93	78	57.4	2.05	25.00	-261.33	54440.65	1	1.82	2.4	4.36	0.01	0.63	-0.93	3.59
chroot.c	77	55.0	2.04	25.93	78	57.4	2.05	25.00	-261.33	54440.65	1	1.82	2.4	4.36	0.01	0.63	-0.93	3.59
cmp.c	132	359.8	2.72	55.22	133	362.6	2.72	55.22	-156.33	43817.04	31	39.24	114.8	46.33	0.16	5.59	-0.45	0.70
col.c	133	1017.1	3.05	122.22	133	1017.1	3.05	122.22	-156.33	43817.04	31	39.24	114.8	46.33	0.16	5.59	-0.45	0.70
col.c	133	1017.1	3.05	122.22	133	1017.1	3.05	122.22	-156.33	43817.04	31	39.24	114.8	46.33	0.16	5.59	-0.45	0.70
comm.c	180	493.1	2.74	108.89	182	516.1	2.84	108.24	-108.33	11732.35	2	1.10	21.0	0.00	0.00	0.00	0.00	0.00
comm.c	180	493.1	2.74	108.89	182	516.1	2.84	108.24	-108.33	11732.35	2	1.10	21.0	0.00	0.00	0.00	0.00	0.00
cpio.c	1051	3539.8	3.37	90.20	1283	4459.1	3.48	110.52	762.67	581672.88	232	22.07	919.3	25.97	0.11	31.18	20.95	22.0
cpio.c	1051	3539.8	3.37	90.20	1283	4459.1	3.48	110.52	762.67	581672.88	232	22.07	919.3	25.97	0.11	31.18	20.95	22.0
cron.c	299	1059.4	3.54	78.59	300	1082.4	3.61	78.33	10.67	113.95	1	0.33	23.0	2.17	0.06	1.83	-0.26	0.33
cron.c	299	1059.4	3.54	78.59	300	1082.4	3.61	78.33	10.67	113.95	1	0.33	23.0	2.17	0.06	1.83	-0.26	0.33
crypt.c	94	183.1	1.95	42.55	95	206.1	2.17	42.10	-194.33	37762.28	1	1.06	23.0	12.56	0.22	11.32	-0.45	1.06
crypt.c	94	183.1	1.95	42.55	95	206.1	2.17	42.10	-194.33	37762.28	1	1.06	23.0	12.56	0.22	11.32	-0.45	1.06
capit.c	436	1126.0	2.58	54.13	439	1151.4	2.62	53.76	147.67	21807.85	3	0.69	25.4	2.26	0.04	1.55	-0.37	0.68
capit.c	436	1126.0	2.58	54.13	439	1151.4	2.62	53.76	147.67	21807.85	3	0.69	25.4	2.26	0.04	1.55	-0.37	0.68
cb.c	1060	3400.8	2.32	80.47	1061	3428.6	2.32	80.47	95.67	9153.67	163	42.45	502.4	39.38	-0.07	2.15	-0.40	0.50
cb.c	1060	3400.8	2.32	80.47	1061	3428.6	2.32	80.47	95.67	9153.67	163	42.45	502.4	39.38	-0.07	2.15	-0.40	0.50
cut.c	123	598.7	2.85	160.98	125	626.1	2.86	161.67	-48.33	17780.33	22	2.63	46.0	19.40	0.69	28.55	10.39	11.09
cut.c	123	598.7	2.85	160.98	125	626.1	2.86	161.67	-48.33	17780.33	22	2.63	46.0	19.40	0.69	28.55	10.39	11.09
date.c	710	1984.7	4.80	127.61	730	2053.9	4.82	127.67	48.33	28.36	2	0.71	-73.4	7.33	-0.26	7.95	0.06	9.05
date.c	710	1984.7	4.80	127.61	730	2053.9	4.82	127.67	48.33	28.36	2	0.71	-73.4	7.33	-0.26	7.95	0.06	9.05
date.c	710	1984.7	4.80	127.61	730	2053.9	4.82	127.67	48.33	28.36	2	0.71	-73.4	7.33	-0.26	7.95	0.06	9.05
deroff.c	283	1001.6	3.54	108.48	285	928.2	3.26	111.93	-55.33	68998.05	4	0.73	46.0	2.94	0.06	2.20	-0.13	0.19
deroff.c	283	1001.6	3.54	108.48	285	928.2	3.26	111.93	-55.33	68998.05	4	0.73	46.0	2.94	0.06	2.20	-0.13	0.19
devnab.c	629	2685.2	4.27	72.77	629	2685.2	4.27	72.77	340.67	116059.32	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00
devnab.c	629	2685.2	4.27	72.77	629	2685.2	4.27	72.77	340.67	116059.32	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00
dr.c	158	552.0	3.49	76.58	158	552.0	3.49	76.58	-251.33	63164.36	5	13.51	2.4	1.58	-0.43	10.49	7.98	10.93
dr.c	158	552.0	3.49	76.58	158	552.0	3.49	76.58	-251.33	63164.36	5	13.51	2.4	1.58	-0.43	10.49	7.98	10.93
echo.c	160	172.3	2.62	31.47	159	173.7	2.69	30.98	-99.33	9865.50	60	37.97	144.3	26.14	-0.30	8.55	-0.43	0.56
echo.c	160	172.3	2.62	31.47	159	173.7	2.69	30.98	-99.33	9865.50	60	37.97	144.3	26.14	-0.30	8.55	-0.43	0.56
env.c	89	300.7	3.32	29.21	92	289.5	3.46	28.37	-228.33	35332.40	3	1.67	2.0	1.39	-0.01	0.97	-2.49	2.50
env.c	89	300.7	3.32	29.21	92	289.5	3.46	28.37	-228.33	35332.40	3	1.67	2.0	1.39	-0.01	0.97	-2.49	2.50
errdead.c	183	662.0	3.62	36.77	172	664.4	3.86	36.37	-105.33	11093.40	-11	6.01	-12.4	0.32	0.21	6.94	2.31	7.91
errdead.c	183	662.0	3.62	36.77	172	664.4	3.86	36.37	-105.33	11093.40	-11	6.01	-12.4	0.32	0.21	6.94	2.31	7.91
errdemon.c	71	270.8	3.81	28.17	75	287.0	3.83	26.67	-217.33	47230.24	4	5.63	16.2	5.98	0.01	0.33	-1.50	5.32
errdemon.c	71	270.8	3.81	28.17	75	287.0	3.83	26.67	-217.33	47230.24	4	5.63	16.2	5.98	0.01	0.33	-1.50	5.32

(continued)

Name	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
crypt.c	1244	3471.9	2.79	42.12	1668	4560.2	2.73	44.18	955.67	913314.31	424	34.08	1088.3	31.35	-0.06	2.03	2.06	4.89
expr.c	412	1442.7	3.50	58.01	356	1165.6	3.27	58.99	123.167	15295.46	-56	13.59	-277.1	19.21	-0.23	6.48	0.98	1.69
freq.c	346	1613.7	4.66	133.24	351	1655.1	4.72	131.34	57.67	3326.38	5	1.45	41.4	2.57	0.05	1.10	-1.90	1.43
find.c	788	3635.1	4.61	57.36	874	3755.0	4.30	71.40	499.67	249674.31	86	10.91	119.9	3.30	-0.32	6.85	14.04	24.47
getch.c	435	934.2	2.14	47.93	1713	6400.8	3.70	72.28	330.33	23615.94	1269	287.12	553.8	59.43	0.66	7.50	-0.64	51.79
getch2.c	435	934.2	2.14	47.93	1713	6400.8	3.70	72.28	330.33	23615.94	1269	287.12	553.8	59.43	0.66	7.50	-0.64	51.79
graph.c	693	3112.7	4.49	58.87	695	3117.3	4.49	58.70	404.67	167661.69	2	0.29	4.6	0.15	-0.01	0.14	-0.17	0.29
grep.c	168	595.6	3.55	61.90	177	646.2	3.65	61.02	-120.33	14478.15	9	5.36	50.6	8.50	0.11	2.97	-0.88	1.42
grpp.c	181	476.5	2.63	8.84	182	478.9	2.63	8.79	-107.33	15187.10	1	0.55	2.4	0.50	0.00	0.05	-0.05	0.56
hp.c	436	1393.2	3.20	40.27	82	1393.2	3.20	47.25	-147.67	31807.85	0	3.00	0.0	0.00	0.00	0.00	0.00	0.00
hypen.c	79	284.4	3.60	101.27	82	288.4	3.64	78.05	209.33	43817.04	3	3.80	14.0	4.92	0.04	1.08	-23.22	22.93
id.c	52	206.6	3.97	72.69	53	209.0	3.94	72.55	-336.33	58494.50	1	1.92	2.4	1.16	-0.03	0.75	-0.14	1.82
id2.c	52	206.6	3.97	72.69	53	209.0	3.94	72.55	-336.33	58494.50	1	1.92	2.4	1.16	-0.03	0.75	-0.14	1.82
join.c	215	774.2	3.60	121.39	218	779.2	3.65	115.72	17.37	5376.38	2809	571.28	12824.2	859.37	1.27	41.22	13.07	24.71
kill.c	59	182.5	3.09	83.05	67	217.1	3.24	67.16	-229.33	52590.05	8	13.56	34.6	18.96	0.15	4.74	-15.89	19.13
kunb.c	234	780.2	3.33	141.45	247	750.4	3.41	136.44	-54.33	2951.23	13	5.56	-29.8	3.82	-0.30	8.86	-5.01	3.54
label.c	116	399.5	3.44	43.97	147	500.7	3.41	43.54	-172.33	29695.98	31	26.72	101.2	25.33	-0.04	1.10	-0.43	0.98
line.c	42	104.9	2.50	9.52	43	107.3	2.50	9.30	-246.33	60676.11	1	2.38	2.4	2.29	0.00	0.09	-0.22	2.31
link.c	262	764.7	2.92	41.22	901	3796.1	4.21	62.15	-26.33	6931.02	639	243.89	3031.4	396.41	1.29	44.20	20.93	50.76
login.c	262	764.7	2.92	41.22	901	3796.1	4.21	62.15	-26.33	6931.02	639	243.89	3031.4	396.41	1.29	44.20	20.93	50.76
logn.c	539	2347.2	4.35	51.35	596	2500.8	4.12	51.80	-29.33	78027.82	5	33.33	4.6	28.73	-0.06	3.42	-2.78	25.00
logn2.c	539	2347.2	4.35	51.35	596	2500.8	4.12	51.80	-29.33	78027.82	5	33.33	4.6	28.73	-0.06	3.42	-2.78	25.00
mail.c	731	2265.1	3.10	68.67	1329	3574.9	2.69	70.35	442.67	195960.98	598	81.88	1309.8	57.83	-0.41	13.15	1.68	2.45
makekey.c	19	2.4	0.13	0.00	20	25.4	1.27	0.00	-269.33	72536.07	1	5.26	23.0	954.36	1.14	839.00	0.00	0.00
manptog.c	26	94.2	3.62	11.54	26	94.2	3.62	11.54	-262.33	68814.51	0	0.00	0.0	0.00	0.00	0.00	0.00	0.00
meag.c	90	246.2	2.74	60.00	91	248.6	2.73	59.34	-198.33	39332.89	1	1.11	2.4	0.97	0.00	0.13	-0.66	1.10
mdir.c	71	144.6	2.04	30.99	72	147.0	2.04	30.56	-217.33	47320.84	1	1.41	2.4	1.66	0.01	0.25	-0.43	1.39
mkfe.c	625	1661.4	2.66	53.92	1117	3164.4	2.83	56.40	336.67	113349.92	492	78.72	1503.0	90.47	0.17	6.55	-2.48	4.60
mkfod.c	96	252.2	4.08	46.37	162	340.2	4.15	42.68	-222.33	49428.50	16	24.24	71.2	26.47	0.07	1.79	-4.29	9.13
mkfod2.c	96	252.2	4.08	46.37	162	340.2	4.15	42.68	-222.33	49428.50	16	24.24	71.2	26.47	0.07	1.79	-4.29	9.13
mw.c	337	856.8	3.62	73.84	344	873.0	3.58	78.49	-52.33	35634.28	17	2.90	15.2	1.89	-0.04	1.64	-0.36	0.54
ncheck.c	365	1491.3	4.09	72.88	415	1700.5	4.10	68.92	76.67	5879.02	50	13.70	209.2	14.03	0.01	0.29	-3.96	5.43
newgr.c	107	358.2	3.35	17.76	130	413.2	3.18	24.61	-181.33	32878.83	23	21.50	55.0	15.35	-0.17	5.04	6.85	38.55
newgr2.c	107	358.2	3.35	17.76	130	413.2	3.18	24.61	-181.33	32878.83	23	21.50	55.0	15.35	-0.17	5.04	6.85	38.55
nice.c	28	103.4	3.69	17.86	41	160.1	3.90	51.22	-369.33	1907.49	1	0.30	2.4	0.23	0.00	0.08	-0.29	0.31
nl.c	453	1283.5	2.83	230.02	474	1453.8	3.07	222.15	164.67	37769.21	13	46.43	56.7	54.83	0.21	5.73	33.36	186.68
nohup.c	250	740.9	3.82	97.50	251	751.4	3.80	68.52	-248.33	61652.51	14	35.00	78.4	107.38	0.98	53.34	-8.98	11.59
pack.c	346	1309.9	3.79	65.61	363	1422.5	3.79	69.15	-57.67	3376.38	17	9.11	112.6	6.62	0.11	3.50	2.51	5.96
pack2.c	339	861.9	3.61	54.39	342	896.3	3.70	57.85	-49.33	2432.98	3	1.26	34.4	3.99	0.10	2.69	-3.46	6.36
paate.c	158	543.6	3.44	174.68	159	566.6	3.56	173.58	-130.33	16984.66	1	0.63	23.0	4.23	0.12	3.57	-1.10	0.63
pcat.c	591	1021.7	3.51	71.48	596	1063.1	3.59	70.27	2.67	7.15	5	1.72	41.4	4.05	0.08	2.29	-1.21	1.69
pr.c	207	2354.2	4.64	84.22	233	2435.2	4.57	86.30	218.67	47818.67	26	5.12	81.0	3.44	-0.07	1.60	2.08	2.47
pr.c2	348	1436.7	4.11	58.31	962	3153.3	3.28	56.24	59.67	35601.08	614	176.44	1716.6	119.48	-0.85	20.55	-2.67	4.53
pr.c3	352	1556.0	4.17	58.31	1189	4420.6	3.72	71.80	653.67	427290.75	246	26.11	871.6	24.56	-0.04	1.23	0.46	0.64
pr.c4	352	1556.0	4.17	58.31	1189	4420.6	3.72	71.80	653.67	427290.75	246	26.11	871.6	24.56	-0.04	1.23	0.46	0.64
pr.c5	176	592.1	3.31	76.14	181	618.9	3.49	70.01	-112.33	1616.95	5	2.88	36.8	6.32	0.11	3.37	-2.11	0.78

(continued)

Names	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
pwd.c	85	276.0	3.25	87.06	90	292.2	3.25	82.22	-203.33	41341.14	5	5.88	16.2	5.87	0.00	0.01	-4.84	2.57
regcomp.c	152	605.1	3.98	111.84	156	646.5	4.14	108.97	-136.33	10584.56	4	2.63	41.4	6.84	0.16	4.09	0.99	2.56
rm.c	155	507.7	3.28	92.90	156	507.7	3.28	92.95	-133.33	17775.61	1	0.65	0.0	0.00	-0.02	0.64	0.05	0.05
rmid.c	104	321.2	3.09	39.42	105	323.6	3.08	39.95	-184.33	33975.78	1	0.96	2.4	0.75	-0.01	0.21	0.64	0.94
setjmp.c	636	1646.9	3.05	37.97	635	1720.6	2.71	87.72	-342.67	118112.37	3	0.97	43.8	2.61	0.06	2.12	-0.25	0.28
setjmp.h	29	56.7	1.96	34.48	30	153.1	1.97	105.35	-52.33	59721.50	9	19.57	20.8	14.76	-4.00	-20.64	16.37	3.33
sleep.c	29	56.7	1.96	34.48	30	153.1	1.97	105.35	-52.33	59721.50	9	19.57	20.8	14.76	-4.00	-20.64	16.37	3.33
sort.c	911	3109.2	3.41	89.02	920	3150.6	3.42	89.24	-622.67	387793.91	9	0.49	2.4	4.23	0.01	0.75	-1.15	0.25
spine.c	333	1124.4	3.38	54.05	334	1147.4	3.44	53.89	-44.67	1995.84	1	0.30	23.0	2.03	0.06	1.77	-0.72	0.44
split.c	81	274.1	3.28	118.52	93	312.9	3.36	113.98	-207.33	49983.74	12	14.81	38.8	14.75	-0.02	0.57	-4.54	3.83
stat.c	588	1344.6	2.39	37.92	663	1609.8	2.43	65.16	-299.67	89804.98	75	12.76	265.2	19.72	0.14	6.15	27.24	71.82
stat.h	160	488.6	3.05	40.00	170	493.2	2.90	45.29	-123.33	16467.36	10	6.25	4.6	0.94	-0.16	4.57	-1.83	1.67
sync.c	75	258.5	0.66	109.33	80	264.1	0.66	109.33	-123.33	45507.64	5	6.67	4.6	1.77	-0.16	4.57	-1.83	1.67
sync.h	75	258.5	0.66	109.33	80	264.1	0.66	109.33	-123.33	45507.64	5	6.67	4.6	1.77	-0.16	4.57	-1.83	1.67
taba.c	720	2463.1	3.42	40.00	706	2712.5	3.40	40.00	-283.33	80273.17	-1	20.00	2.4	24000.00	0.40	4000.00	0.00	0.00
taba.h	487	1768.3	3.63	76.59	489	1791.3	3.66	76.28	-198.67	139471.67	2	0.47	36.8	5.11	0.09	0.98	-0.31	0.40
tail.c	187	719.6	3.85	83.96	192	756.4	3.94	81.77	-101.33	10266.80	5	0.11	2.4	0.08	0.00	0.02	-0.08	0.11
tar.c	939	2857.2	3.04	77.42	940	2859.6	3.04	77.34	-650.67	423377.69	1	0.94	21.0	0.00	0.00	0.00	0.00	0.00
tc.c	643	1900.2	2.96	65.01	643	1900.2	2.96	65.01	-354.67	125.00	0	94.21	0.0	0.00	0.00	0.00	0.00	0.00
tc.h	91	376.9	4.14	76.92	92	379.3	4.12	76.09	-197.33	38937.24	1	1.10	2.4	0.64	-0.02	0.46	-0.83	1.08
tc.h	91	376.9	4.14	76.92	92	379.3	4.12	76.09	-197.33	38937.24	1	1.10	2.4	0.64	-0.02	0.46	-0.83	1.08
touch.c	188	822.1	4.95	68.23	188	822.1	4.95	68.23	-100.33	10085.15	-10	5.32	-115.0	13.98	-0.40	9.13	2.73	3.98
tr.c	144	681.1	4.73	92.36	148	685.7	4.93	89.16	-182.33	6777.46	1	0.78	3.6	0.68	-0.10	2.04	-2.50	2.71
taort.c	206	957.4	4.65	36.89	207	959.8	4.64	36.71	-182.33	6777.46	1	0.78	3.6	0.68	-0.10	2.04	-2.50	2.71
tty.c	18	32.2	1.79	11.11	44	209.1	1.75	59.09	-270.33	7375.72	26	14.44	176.9	54.21	2.91	16.72	7.68	43.30
unmount.c	49	172.5	3.52	44.90	79	256.7	3.25	46.83	-239.33	7375.72	30	61.22	84.2	48.81	-0.27	7.68	43.30	4.30
uname.c	52	252.7	4.86	38.46	65	283.5	4.36	36.92	-236.33	55849.60	13	25.00	30.8	12.19	-0.50	10.23	-1.54	4.00
uniq.c	142	509.9	3.59	65.49	148	514.5	3.48	65.54	-146.33	21411.07	6	4.23	4.6	0.90	-0.11	3.18	0.05	0.08
uniq.h	465	2109.6	4.53	47.53	467	2165.4	4.64	47.32	-176.67	31213.98	2	0.43	59.8	2.84	0.11	2.39	-0.21	0.44
unpack.c	301	1200.4	3.99	77.08	305	1117.8	1.53	22.22	-280.33	78582.22	1	12.50	2.4	21.03	0.11	7.55	-0.28	11.12
unpack.h	568	1620.4	4.61	57.04	569	2622.8	4.61	55.96	-122.67	78582.22	64	21.26	-22.6	1.88	-0.76	19.04	6.48	8.41
vix.c	301	1200.4	3.99	77.08	305	1117.8	1.53	22.22	-280.33	78582.22	1	12.50	2.4	21.03	0.11	7.55	-0.28	11.12
volcopy.c	412	1125.1	2.73	67.23	916	2729.0	2.98	72.60	-123.67	15295.46	504	122.33	1603.9	141.53	0.09	0.08	-0.10	0.18
wc.c	113	300.7	2.66	99.11	115	305.3	2.65	97.39	-175.33	30738.93	2	1.77	4.6	1.53	-0.01	0.26	5.37	7.99
who.c	66	250.6	3.80	60.61	638	2570.6	4.03	75.55	-222.33	49438.50	572	866.67	2320.0	925.74	0.23	6.10	14.94	24.65
write.c	189	669.4	3.54	55.03	313	886.3	2.83	48.88	-99.33	9865.50	124	65.61	216.9	32.40	-0.71	19.99	-6.15	11.17
xarg.c	455	1718.1	3.78	54.94	436	1658.6	3.80	60.32	-166.67	27780.49	-19	4.18	-59.5	3.46	0.03	0.74	5.38	9.79
Average	288	976.3	3.26	67.56	374	1304.7	3.33	68.99	3.0E-14	73554.66	86	32.72	328.4	244.93	0.07	46.32	1.44	20.35
Stddev	273	965.0	0.86	42.15	448	1730.4	0.78	40.40	272.52	145161.85	309	104.55	1324.2	2165.48	0.43	367.59	9.40	1173.15
Maximum	1255	5351.3	4.86	230.02	3295	14318.2	4.96	222.15	966.67	934460.16	2809	866.67	1284.2	24000.00	2.96	4000.00	47.98	113.01
Minimum	5	0.0	0.00	0.00	6	2.4	0.40	0.00	-283.33	7.15	-72	0.00	-277.1	0.00	-0.85	0.00	-23.22	0.00
Total	35464	120091	401.4	8309.4	46023	160479	409.5	8486.0	3.0E-12	9060752.9	10559	4024.6	40387.8	30126.83	8.17	5697.43	176.51	2503.53

APPENDIX C. STATEMENT TYPES

Statement Types (123 modules)		1. for	2. while	3. if	4. else	5. switch	6. case	7. goto	8. break	9. assignments	10. preprocessors	11. comments	12. blanklines	13. return	14. input	15. output	16. functions	17. declarations	18. default
Names	System3,5	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
300.c	sys3	3	7	44	14	0	0	2	2	88	23	75	33	1	3	6	79	23	0
	-- 5	3	7	44	14	0	0	2	2	88	23	75	33	1	3	6	79	23	0
300s.c	sys3	3	7	45	14	0	0	2	2	86	23	76	31	1	3	8	112	23	0
	-- 5	3	7	45	14	0	0	2	2	86	23	76	31	1	3	8	79	22	0
4014.c	sys3	0	4	25	2	4	18	0	4	96	19	1	25	3	7	11	102	32	3
	-- 5	0	4	25	3	4	18	0	4	97	19	1	25	3	7	11	105	32	3
450.c	sys3	2	5	27	6	1	4	2	4	59	24	69	27	1	3	4	91	20	0
	-- 5	2	5	27	6	1	4	2	4	59	24	69	27	1	3	4	91	20	0
arvc.c	sys3	3	2	6	0	0	0	2	0	20	2	0	8	4	0	4	36	8	0
	-- 5	1	7	27	1	1	1	0	2	36	6	69	116	0	0	0	171	16	1
banner.c	sys3	7	1	5	0	0	0	0	0	21	7	2	9	0	0	3	19	10	7
	-- 5	7	1	5	0	0	0	0	0	22	8	2	9	0	0	3	22	12	0
bcopy.c	sys3	0	3	13	0	0	0	9	1	15	0	0	8	0	0	8	28	9	0
	-- 5	0	3	13	0	0	0	9	1	15	4	1	9	0	0	8	28	8	0
bfs.c	sys3	15	27	162	39	6	52	1	27	283	1	5	222	5	34	13	502	152	3
	-- 5	15	27	162	39	6	52	1	27	286	4	7	223	5	34	13	506	155	3
cal.c	sys3	6	3	13	0	1	2	4	3	46	0	9	26	0	0	13	26	9	1
	-- 5	6	3	13	0	1	2	4	3	47	1	9	26	0	0	13	27	10	1
cat.c	sys3	1	2	17	1	1	3	0	3	16	5	1	4	0	1	1	26	8	0
	-- 5	1	2	19	1	1	3	0	3	18	10	1	5	0	1	1	31	9	0
cb.c	sys3	3	9	53	6	1	15	5	0	140	2	0	2	0	1	10	107	27	1
	-- 5	4	26	185	59	3	25	3	3	241	56	8	15	1	3	9	415	64	2
checkcw.c	sys3	2	5	25	3	2	4	0	6	41	1	4	19	0	0	9	27	9	1
	-- 5	2	5	25	3	2	4	0	6	42	2	4	19	0	0	9	28	10	1
checkeq.c	sys3	2	3	21	8	1	1	0	2	22	1	1	4	0	0	14	19	2	0
	-- 5	2	3	21	8	1	1	0	2	23	2	1	4	0	0	14	20	3	0
chgrp.c	sys3	1	1	5	1	0	0	0	0	6	5	1	6	0	0	0	18	6	0
	-- 5	1	1	5	1	0	0	0	0	6	5	2	6	0	0	0	18	6	0
chmod.c	sys3	3	3	6	0	5	18	2	3	36	12	10	14	4	0	0	35	9	2
	-- 5	3	3	6	0	5	18	2	3	36	12	11	14	4	0	0	35	9	2
chown.c	sys3	1	1	5	0	0	0	1	0	6	5	1	7	0	0	0	18	6	0
	-- 5	1	1	5	0	0	0	1	0	6	5	2	7	0	0	0	18	6	0
chroot.c	sys3	0	0	4	0	0	0	0	0	2	1	2	1	0	0	3	13	0	0
	-- 5	0	0	4	0	0	0	0	0	2	1	3	1	0	0	3	13	0	0
clri.c	sys3	3	1	7	0	0	0	0	0	14	5	1	5	0	0	6	31	6	0
	-- 5	3	1	11	3	0	0	0	0	18	11	3	8	0	0	8	41	8	0
cmp.c	sys3	0	4	21	1	0	0	0	0	24	2	1	13	0	4	2	32	4	0
	-- 5	0	4	21	1	0	0	0	0	25	3	1	13	0	4	2	33	5	0
col.c	sys3	4	9	26	5	4	19	0	11	57	9	6	39	1	2	10	53	23	4
	-- 5	4	9	26	5	4	19	0	11	57	9	6	39	1	2	10	53	23	4
comm.c	sys3	0	5	21	1	3	9	0	5	19	2	1	23	3	1	1	48	7	1
	-- 5	0	5	21	1	3	9	0	5	22	3	1	23	3	1	1	49	8	1
cpio.c	sys3	11	13	132	18	4	24	16	20	175	48	11	68	30	0	8	351	68	1
	-- 5	20	18	166	25	4	28	7	30	240	58	68	83	27	0	9	439	85	1
cron.c	sys3	5	8	37	5	1	4	13	1	65	13	1	26	3	9	0	88	16	0
	-- 5	5	8	37	5	1	4	13	1	66	14	1	26	3	9	0	89	17	0
crypt.c	sys3	4	3	6	1	0	0	0	0	29	4	1	6	0	0	1	32	2	0
	-- 5	4	3	6	1	0	0	0	0	30	5	1	6	0	0	1	33	3	0
csplit.c	sys3	10	6	32	7	3	10	0	9	37	18	28	40	1	0	0	109	22	2
	-- 5	10	6	32	7	3	10	0	9	38	19	29	41	1	0	0	111	23	2
ct.c	sys3	4	9	53	6	1	5	1	7	54	20	1	34	6	2	0	174	23	0
	-- 5	8	10	59	5	1	5	0	8	89	19	26	78	3	8	0	308	42	0
cuc.c	sys3	13	11	102	20	2	12	8	19	143	61	70	51	9	0	10	404	23	2
	-- 5	15	10	88	29	1	7	2	17	118	64	88	46	9	0	0	334	59	1
cut.c	sys3	4	5	25	7	2	7	0	6	43	4	8	5	0	0	0	27	8	2
	-- 5	4	5	25	7	2	7	0	6	48	5	8	5	0	0	0	28	9	2
cw.c	sys3	2	14	84	39	2	9	1	14	116	6	168	20	8	2	56	192	21	2
	-- 5	2	14	87	41	2	9	1	14	121	7	173	21	8	2	57	191	22	2

(continued)

Names	System3,5	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
date.c	sys3	4	2	25	2	2	19	0	0	84	12	6	25	1	0	0	73	26	1
-- 5		4	2	25	2	2	19	0	0	86	15	8	27	1	0	0	95	21	1
dd.c	sys3	4	9	66	6	1	6	10	0	98	13	5	37	5	0	0	109	26	0
-- 5		4	9	67	6	1	6	10	0	99	16	6	37	5	0	0	112	27	0
deroff.c	sys3	13	28	95	52	2	19	7	4	106	17	12	90	15	3	10	106	52	1
-- 5		13	28	95	52	2	19	7	4	106	17	12	90	15	3	10	106	52	1
devnm.c	sys3	0	2	6	0	0	0	1	0	2	4	0	3	0	0	0	1	17	2
-- 5		0	2	6	0	0	0	1	0	2	4	1	3	0	0	0	1	19	2
df.c	sys3	5	2	16	2	1	3	1	3	23	8	1	12	2	0	7	54	13	1
-- 5		6	1	24	6	1	3	0	4	37	12	5	17	5	0	11	64	15	1
du.c	sys3	5	2	29	2	1	3	0	1	37	11	9	19	1	0	3	52	12	1
-- 5		5	2	29	2	1	3	0	1	37	17	10	20	1	0	3	55	12	1
echo.c	sys3	2	1	2	0	1	8	0	0	5	0	0	8	0	0	10	6	4	1
-- 5		2	1	2	0	1	8	0	0	5	0	1	8	0	0	10	6	4	1
env.c	sys3	2	3	7	1	0	0	0	0	11	2	2	13	2	0	1	23	9	0
-- 5		2	3	7	1	0	0	0	0	10	2	5	13	0	0	1	34	8	0
errdead.c	sys3	6	0	18	1	0	0	0	0	24	21	0	9	0	0	1	80	17	0
-- 5		6	0	18	1	0	0	0	0	24	21	1	6	0	0	1	80	17	0
errdemon.c	sys3	1	1	9	1	0	0	0	0	12	9	0	5	0	0	0	36	6	0
-- 5		1	1	9	1	0	0	0	0	12	12	1	5	0	0	0	36	6	0
errpt.c	sys3	16	8	87	20	3	20	1	12	156	49	32	60	23	0	100	336	100	3
-- 5		19	8	101	30	6	34	1	17	240	114	44	70	35	0	120	350	126	6
expr.c	sys3	3	4	33	7	6	35	2	21	79	31	0	36	21	0	0	119	33	0
-- 5		1	1	31	7	6	29	2	18	67	28	0	26	17	0	0	122	24	0
fgrep.c	sys3	3	7	66	17	1	9	15	5	84	4	1	22	3	1	9	58	29	0
-- 5		3	7	66	17	1	9	15	5	85	9	1	22	3	1	9	59	30	0
find.c	sys3	7	13	93	26	2	6	13	3	131	8	27	45	5	4	0	468	115	0
-- 5		7	9	106	35	1	4	6	6	129	29	29	50	5	1	0	460	115	1
getopt.c	sys3	0	2	4	0	0	0	0	0	8	1	0	10	0	0	1	16	5	0
-- 5		0	2	4	0	0	0	0	0	8	1	0	10	0	0	1	16	7	0
getty.c	sys3	5	2	28	6	1	2	1	3	66	11	31	47	0	0	0	97	22	1
-- 5		20	14	110	52	5	26	0	28	269	116	343	248	3	3	7	363	185	3
graph.c	sys3	12	6	74	16	2	20	3	18	171	6	2	55	11	5	0	195	105	2
-- 5		12	6	74	16	2	20	3	18	172	7	2	56	11	5	0	196	105	2
grep.c	sys3	1	3	16	1	1	7	0	7	14	11	3	21	5	1	7	41	15	0
-- 5		1	3	16	2	1	7	0	7	16	14	3	22	5	1	7	44	17	0
grpck.c	sys3	5	2	21	8	0	0	0	0	27	9	12	22	4	0	2	11	6	0
-- 5		5	2	21	8	0	0	0	0	27	9	13	22	4	0	2	11	6	0
hp.c	sys3	4	12	39	13	1	5	0	10	84	23	97	24	3	1	3	130	24	1
-- 5		4	12	39	13	1	5	0	10	84	23	97	24	3	1	3	130	24	1
hyphen.c	sys3	0	4	10	0	0	0	4	0	26	0	0	9	0	0	1	25	5	0
-- 5		0	4	9	0	0	0	4	0	27	1	1	9	0	0	1	27	6	0
id.c	sys3	0	0	4	0	0	0	0	0	8	3	0	8	0	0	5	20	8	0
-- 5		0	0	4	0	0	0	0	0	8	3	1	8	0	0	5	20	8	0
init.c	sys3	18	12	39	11	3	7	0	8	87	28	4	72	3	2	1	198	33	1
-- 5		35	27	174	56	6	52	0	55	378	277	771	453	11	0	0	760	455	5
join.c	sys3	6	6	22	12	2	7	0	10	46	7	19	12	0	0	7	66	16	1
-- 5		6	6	22	12	2	7	0	10	47	8	19	14	0	0	7	67	17	1
kill.c	sys3	0	1	9	4	0	0	0	0	15	3	1	3	0	0	0	17	3	0
-- 5		0	1	9	4	0	0	0	0	15	10	2	3	0	0	0	17	3	0
kunb.c	sys3	2	2	27	8	4	20	8	9	26	2	0	9	0	0	39	48	11	2
-- 5		2	2	27	8	4	20	8	9	26	15	1	9	0	0	39	59	6	2
labelit.c	sys3	2	0	16	1	0	0	3	0	6	8	1	9	0	0	6	47	7	0
-- 5		2	0	17	2	0	0	3	0	7	23	2	11	0	0	8	66	8	0
line.c	sys3	0	1	3	1	0	0	0	0	8	1	1	2	1	0	0	11	3	0
-- 5		0	1	3	1	0	0	0	0	8	1	2	2	1	0	0	11	3	0
link.c	sys3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0
-- 5		0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0
login.c	sys3	1	3	22	2	0	0	3	0	40	28	10	14	0	1	11	98	16	0
-- 5		10	13	73	10	3	27	6	15	128	55	108	126	3	3	18	285	110	3
logname.c	sys3	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	5	0	0
-- 5		0	0	1	0	0	0	0	0	1	2	2	0	0	0	0	7	0	0
ls.c	sys3	9	5	54	22	2	18	0	5	98	14	40	47	1	0	18	118	73	1
-- 5		9	5	58	23	2	18	0	6	116	26	52	52	1	0	18	143	78	1
mail.c	sys3	14	11	87	14	2	22	2	14	142	17	17	44	11	2	6	344	43	1
-- 5		18	10	123	11	3	26	3	19	180	57	97	81	15	2	16	435	68	1
makekey.c	sys3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0
-- 5		0	0	0	0	0	0	0	0	1	1	1	3	0	0	0	8	1	0

(continued)

Names	System	3	5	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
manprog.c	sys3	0	0	2	0	0	0	0	0	0	0	1	3	1	2	0	0	1	7	3	0
-- 5		0	0	2	0	0	0	0	0	0	0	1	3	1	2	0	0	1	7	3	0
mesg.c	sys3	1	1	8	1	2	5	0	5	8	3	4	10	0	0	0	0	2	21	4	1
-- 5		1	1	8	1	2	5	0	5	8	3	5	10	0	0	0	0	2	21	4	1
mkdir.c	sys3	1	1	7	0	0	0	0	0	0	6	2	1	7	3	0	0	0	36	2	0
-- 5		1	1	7	0	0	0	0	0	0	6	2	2	7	3	0	0	0	36	2	0
mkfs.c	sys3	28	5	39	2	3	13	5	9	164	14	17	70	1	1	22	145	38	1	1	
-- 5		41	10	85	5	3	18	9	17	285	85	37	112	1	1	30	255	59	1	1	
mknod.c	sys3	0	2	11	3	0	0	0	0	0	10	3	1	4	0	0	0	28	6	0	0
-- 5		0	2	11	3	0	0	0	0	0	10	3	1	4	0	0	0	28	6	0	0
mount.c	sys3	4	4	14	0	0	0	0	0	0	15	5	1	7	1	0	2	38	4	0	0
-- 5		4	4	16	0	0	0	0	0	0	17	9	2	7	1	0	2	43	4	0	0
mv.c	sys3	2	4	42	3	0	0	2	0	26	10	1	19	6	2	0	117	14	0	0	
-- 5		2	4	41	3	0	0	2	0	27	16	1	22	6	2	0	118	14	0	0	
ncheck.c	sys3	15	4	45	2	1	3	2	14	54	16	3	29	9	1	8	105	38	1	1	
-- 5		15	4	49	5	1	3	2	14	63	34	6	32	11	1	9	125	42	1	1	
newgrp.c	sys3	1	1	11	0	0	0	0	0	0	13	6	2	12	6	0	0	42	10	0	0
-- 5		1	1	14	1	0	0	0	0	0	20	6	3	12	6	0	0	53	11	0	0
news.c	sys3	5	6	27	7	2	8	1	8	36	8	11	39	2	1	18	102	29	2	2	
-- 5		5	6	27	7	2	8	1	8	36	8	12	39	2	1	18	102	29	2	2	
nice.c	sys3	0	0	2	0	0	0	0	0	0	2	1	1	3	0	0	0	8	4	0	0
-- 5		0	0	1	4	0	0	0	0	0	3	2	2	5	0	0	0	11	5	0	0
nl.c	sys3	4	7	28	19	6	34	0	35	121	10	26	25	1	0	14	65	29	5	5	
-- 5		5	7	32	19	6	35	0	36	130	10	34	27	1	0	10	62	34	5	5	
nohup.c	sys3	0	0	6	0	0	0	0	0	0	3	1	0	2	0	0	0	26	0	0	0
-- 5		0	0	6	1	0	0	0	0	0	5	1	2	5	0	0	0	27	4	0	0
od.c	sys3	8	2	26	6	3	17	1	17	48	3	1	17	2	0	18	52	12	1	1	
-- 5		8	2	27	7	3	19	1	19	52	5	2	18	2	0	19	59	12	1	1	
pack.c	sys3	14	6	29	2	0	0	5	1	96	13	32	26	9	0	22	75	34	0	0	
-- 5		14	6	32	2	0	0	5	1	99	21	34	27	9	0	23	82	36	0	0	
passwd.c	sys3	2	3	34	9	0	0	18	0	43	6	9	24	0	0	0	89	10	0	0	
-- 5		2	3	35	9	0	0	18	0	44	7	9	24	0	0	0	89	10	0	0	
paste.c	sys3	3	5	20	7	2	5	0	7	55	5	8	8	0	2	0	26	9	2	2	
-- 5		3	5	20	7	2	5	0	7	56	6	8	8	0	2	0	26	9	2	2	
pcat.c	sys3	9	2	33	2	0	0	8	0	82	17	17	22	12	0	0	60	19	0	0	
-- 5		9	2	33	2	0	0	8	0	83	22	17	23	12	0	0	61	20	0	0	
pr.c	sys3	13	7	85	15	6	37	0	12	168	29	21	44	21	4	10	187	43	3	3	
-- 5		13	7	90	16	6	38	0	13	175	29	21	46	22	4	10	195	44	3	3	
prof.c	sys3	11	5	42	4	0	0	1	3	78	13	9	18	0	0	5	127	40	0	0	
-- 5		7	5	55	5	1	13	3	9	126	106	248	165	10	0	0	325	67	0	0	
ps.c	sys3	13	11	111	28	1	13	0	21	180	41	91	67	7	0	40	294	87	0	0	
-- 5		21	13	125	33	1	13	0	25	194	106	107	96	2	0	40	424	103	0	0	
ptx.c	sys3	8	16	54	5	4	15	0	14	95	14	21	79	1	3	6	150	28	4	4	
-- 5		8	16	54	5	4	15	0	14	96	15	21	79	1	3	6	151	29	4	4	
pwck.c	sys3	8	1	22	6	0	1	0	2	30	14	10	24	2	0	0	54	9	0	0	
-- 5		8	1	22	6	0	1	0	2	31	18	10	25	2	0	0	55	10	0	0	
pwd.c	sys3	3	3	10	1	0	0	0	0	12	4	1	8	0	0	0	39	5	0	0	
-- 5		3	3	10	1	0	0	0	0	12	7	2	9	0	0	0	39	5	0	0	
regcmp.c	sys3	2	16	16	5	2	7	1	3	43	1	0	2	0	2	5	58	11	2	2	
-- 5		2	16	16	5	2	7	1	3	45	2	0	4	0	2	5	59	13	2	2	
rm.c	sys3	0	5	24	3	1	3	0	3	10	4	1	13	6	2	5	53	8	1	1	
-- 5		0	5	24	3	1	3	0	3	10	4	1	13	6	2	5	57	8	1	1	
rmdir.c	sys3	0	2	14	0	0	0	2	0	7	4	5	6	8	0	0	48	5	0	0	
-- 5		0	2	14	0	0	0	2	0	7	4	5	6	8	0	0	48	5	0	0	
sdiff.c	sys3	2	22	56	6	5	22	0	20	111	12	22	84	0	0	11	178	31	4	4	
-- 5		2	22	56	6	5	22	0	20	114	13	23	84	0	0	11	179	33	4	4	
setamt.c	sys3	4	1	4	0	0	0	0	2	11	4	0	2	0	0	0	11	2	0	0	
-- 5		4	1	4	0	0	0	0	2	11	8	1	6	0	0	0	11	2	0	0	
sleep.c	sys3	0	1	2	0	0	0	0	0	0	4	1	1	3	0	0	0	6	1	0	0
-- 5		0	1	2	0	0	0	0	0	0	4	1	2	3	0	0	0	6	1	0	0
sort.c	sys3	17	35	103	18	2	16	8	15	176	12	2	70	4	4	2	210	68	2	2	
-- 5		17	35	103	19	2	16	8	15	177	12	2	71	4	4	2	217	69	2	2	
spline.c	sys3	8	1	29	2	3	14	1	11	80	3	4	48	0	2	4	63	32	1	1	
-- 5		8	1	29	2	3	14	1	11	81	4	4	48	0	2	4	64	33	1	1	
split.c	sys3	3	1	9	3	1	11	1	0	21	1	0	6	0	1	1	13	3	0	0	
-- 5		3	1	12	4	1	11	1	0	24	2	0	6	0	1	1	22	3	0	0	
stty.c	sys3	6	1	77	22	1	2	1	0	73	3	0	28	2	0	94	225	20	1	1	
-- 5		6	1	94	31	1	2	1	0	88	6	1	31	2	0	111	241	23	1	1	

(continued)

Names	System3,5	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
su.c	sys3	1	0	20	4	0	0	1	1	40	16	1	9	1	0	1	76	9	0
-- 5		1	0	21	4	0	0	1	1	43	14	2	9	1	0	1	84	9	0
sum.c	sys3	0	3	8	4	0	0	0	0	18	2	2	4	0	2	4	24	7	0
-- 5		0	3	8	4	0	0	0	0	21	3	2	5	0	2	4	28	7	0
sync.c	sys3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
-- 5		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
sysdef.c	sys3	11	4	52	5	4	10	0	8	94	35	36	66	0	0	49	327	77	4
-- 5		12	1	53	7	2	7	0	5	94	99	35	57	0	0	61	308	77	2
tabs.c	sys3	6	14	38	19	3	13	1	13	129	22	87	30	1	0	0	84	41	0
-- 5		6	14	38	19	3	13	1	13	130	23	87	31	1	0	0	85	42	0
tail.c	sys3	2	11	27	4	1	4	7	4	32	6	1	9	0	0	0	38	9	1
-- 5		2	11	27	4	1	4	7	4	33	10	1	10	0	0	0	39	10	1
tar.c	sys3	21	14	103	13	1	20	5	18	133	21	2	90	9	2	10	323	61	1
-- 5		21	14	103	13	1	20	5	18	133	21	3	90	9	2	10	323	61	1
tc.c	sys3	1	8	42	6	5	32	5	13	150	12	4	14	4	1	1	157	55	3
-- 5		1	8	42	6	5	32	5	13	150	12	4	14	4	1	1	157	55	3
tee.c	sys3	4	3	6	1	1	3	0	2	15	5	1	7	0	0	0	21	11	0
-- 5		4	3	6	1	1	3	0	2	15	5	2	7	0	0	0	21	11	0
time.c	sys3	1	2	6	0	0	0	0	0	14	3	2	7	0	0	2	25	8	0
-- 5		1	2	6	0	0	0	0	0	15	5	4	7	0	0	2	26	9	0
touch.c	sys3	2	3	25	4	1	4	0	3	42	5	1	16	1	0	0	57	24	0
-- 5		2	3	25	4	1	4	0	3	44	8	1	16	1	0	0	70	17	0
tr.c	sys3	5	6	23	3	2	5	2	1	61	1	1	9	0	0	1	25	14	1
-- 5		5	6	23	3	2	5	2	1	62	2	1	11	0	0	1	27	14	1
tsort.c	sys3	10	2	20	0	0	0	0	4	41	1	9	13	1	0	1	54	33	0
-- 5		10	2	20	0	0	0	0	4	41	1	10	13	1	0	1	54	33	0
tty.c	sys3	0	0	1	1	0	0	0	0	1	0	1	3	0	0	1	6	1	0
-- 5		0	1	3	1	1	3	0	2	5	2	2	6	0	0	4	10	7	0
umount.c	sys3	1	4	5	0	0	0	0	0	8	4	0	3	0	0	0	23	3	0
-- 5		1	4	5	0	1	6	0	6	8	5	1	3	0	0	0	31	5	1
uname.c	sys3	0	1	9	0	1	6	0	5	7	2	0	0	0	0	4	10	5	0
-- 5		0	1	11	0	1	7	0	6	8	2	1	4	0	0	5	12	5	0
uniq.c	sys3	1	8	14	1	1	3	0	3	15	2	1	15	2	1	2	39	13	0
-- 5		1	8	14	2	1	3	0	3	17	3	1	16	2	1	2	42	13	0
units.c	sys3	12	8	57	3	1	14	28	1	111	3	0	33	1	1	19	66	48	0
-- 5		12	8	57	3	1	14	28	1	112	4	0	33	1	1	19	70	51	0
unlink.c	sys3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0
-- 5		0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	5	0	0
unpack.c	sys3	9	2	34	3	0	0	10	0	78	17	16	22	12	0	16	55	27	0
-- 5		9	4	43	4	0	0	10	0	88	20	20	35	13	0	1	75	18	0
vlx.c	sys3	31	4	42	3	4	17	1	13	111	10	19	38	4	0	10	133	89	1
-- 5		31	4	42	3	4	17	1	13	111	10	20	38	4	0	10	133	89	1
volcopy.c	sys3	7	1	58	10	0	0	7	0	65	16	4	35	8	0	20	179	13	0
-- 5		10	5	130	27	2	5	7	2	140	81	34	63	7	0	46	357	30	2
wc.c	sys3	1	2	10	2	1	3	0	4	23	1	2	12	0	0	6	12	7	1
-- 5		1	2	10	2	1	3	0	4	24	2	2	13	0	0	6	13	7	1
who.c	sys3	1	1	7	1	0	0	0	1	6	4	3	5	0	0	0	25	7	0
-- 5		11	7	49	17	1	13	0	14	94	18	104	114	1	0	0	168	79	1
write.c	sys3	4	2	26	1	0	0	6	2	21	7	2	16	0	0	1	75	12	0
-- 5		1	2	19	10	0	0	0	0	20	8	54	68	1	0	0	102	26	0
xargs.c	sys3	4	15	44	10	3	17	0	18	133	15	11	46	15	0	0	43	46	3
-- 5		3	13	44	10	3	17	0	18	128	12	10	41	15	0	0	51	45	3
TOTAL		1. 592/681				7. 273/248				13. 359/389									
		2. 624/695				8. 598/753				14. 124/132									
(sys3/sys5)		3. 3771/4542				9. 6714/8117				15. 889/971									
		4. 739/1010				10. 1181/2332				16. 10479/13260									
		5. 156/175				11. 1418/3419				17. 2693/3749									
		6. 829/992				12. 2901/4288				18. 89/105									

AN ANALYSIS OF CHANGES DURING
MAINTENANCE OF A C SOFTWARE SYSTEM

by

KYUNG HEE AN

B.S., Korea University, 1977

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1987

ABSTRACT

Software maintenance of computer systems has been an important task. Current demands require the development of good tools for evaluating software during maintenance and enhancement. The maintenance process is not well understood so far. The first step of my research analyzed the relation of changes between the Unix System3 and System5 of C modules. The analysis will help evaluating and identifying changes within modules. One goal of this research is the development of a measure to predict where software changes are likely to occur.

The result section of the paper describes the relationships among several predictors such as lines of code, weight, and nesting levels. The concluding section represents the evaluation of the predictors.